ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, Illinois 60439

Gradient Projection for General Quadratic Programs[1]

Michael P. Friedlander and Sven Leyffer

Mathematics and Computer Science Division

Preprint ANL/MCS-P1370-0906

September 15, 2006

[1]This work was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, Office of Science, U.S. Department of Energy, under Contract W-31-109-ENG-38, and by the National Science and Engineering Research Council of Canada.

Contents

Gradient Projection for General Quadratic Programs[*]

Michael P. Friedlander[†] Sven Leyffer[‡]

September 15, 2006

Abstract

We present a hybrid algorithm for solving large-scale quadratic programs (QPs) that is based on a combination of techniques from gradient projection, augmented Lagrangian, and filter methods. The resulting algorithm is globally and finitely convergent. The method efficiently accommodates a matrix-free implementation and is suitable for large-scale problems with many degrees of freedom. The algorithm is based on two main phases. First, gradient projection iterations approximately minimize the augmented Lagrangian function and provide an estimate of the optimal active set. Second, an equality-constrained QP is approximately minimized on this subspace in order to generate a second-order search direction. Numerical experiments on a subset of the CUTEr QP test problems demonstrate the effectiveness of the proposed approach.

Keywords: Large-scale optimization, quadratic programming, gradient-projection methods, active-set methods, filter methods, augmented Lagrangian.

AMS-MSC2000: 90C20, 90C52.

1 Introduction

Quadratic programs (QPs) play a fundamental role in optimization. They are useful across a rich class of applications, such as the simulation of rigid multibody dynamics [1, 49, 62], optimal control [6, 17, 32, 52, 55], and financial-portfolio optimization [12, 53, 54, 60, 67]. They also arise as a sequence of subproblems within algorithms for solving more general nonlinear optimization problems. Of particular interest for us are sequential quadratic programming (SQP) methods, which have proved a reliable approach for general problems. (For a recent survey, see Gould and Toint [46].) Our purpose is to develop a QP algorithm that may be used effectively within an SQP framework for large-scale nonlinear problems.

[*]Preprint ANL/MCS-P1370-0906 and UBC Department of Computer Science Tech. Rep. TR-2006-18.

[†]Department of Computer Science, University of British Columbia, Vancouver, BC V6T 1Z4, Canada, `mpf@cs.ubc.ca`. The research of this author was supported in part by a Discovery Grant from the National Science and Engineering Research Council of Canada.

[‡]Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439, `leyffer@mcs.anl.gov`. The work of this author was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, Office of Science, U.S. Department of Energy, under Contract W-31-109-ENG-38

Relative to interior-point methods for QPs, active-set methods are especially effective as subproblem solvers within the SQP framework because they can exploit good starting points in order to reduce the number of iterations required for convergence. Inertia-controlling active-set strategies (see, e.g., [33, 41]) are robust in practice, but their overall efficiency is limited by the number of active-set changes that can be made at each iteration. (Typically, a single index changes at each iteration). The combinatorial nature of such an approach severely limits the applicability of inertia-controlling methods to truly large-scale, nonconvex problems that may have many degrees of freedom. However, the robustness and warm-start capability of active-set approaches motivate us to propose a method that is capable of extremely large changes to the active set at each iteration and yet continues to be finitely convergent.

Interior-point methods are often preferred over active-set approaches because they have proved effective for large problems and because they have strong theoretical convergence properties. (For convex QPs, interior methods are convergent in polynomial time [58].) However, the key subproblems within these methods lead to linear systems (known as Karush-Kuhn-Tucker, or KKT, systems) that are inherently ill-conditioned. (See, e.g., [37, theorem 4.2]). Implementations based on iterative linear solvers need to overcome this ill-conditioning by appealing to specialized preconditioners; this has led to significant research efforts for effective preconditioners. (Work on such specialized preconditioners includes [42], and more recently, [9].)

In contrast, the KKT systems that arise in active-set methods do not suffer from artificial ill-conditioning; these systems are therefore more amenable to "standard" preconditioning techniques. We are particularly interested in developing methods that have a strong potential to be effective within a matrix-free context and that can be applied to the large problems arising in PDE-constrained optimization.

We propose a new algorithm for solving QPs that is motivated by the computational effectiveness of gradient-projection methods for bound-constrained QPs. A simplistic extension of gradient projection to general QPs would lead to a subproblem that is almost as difficult to solve as the original QP: each projection of the objective gradient onto the feasible set is itself a QP (see, e.g., [21]). Instead, we use the augmented Lagrangian function to temporarily transform the QP into a bound-constrained problem on which we can perform inexpensive gradient-projection steps.

Each iteration of our algorithm has two main phases. The first phase applies inexpensive gradient-projection iterations in order to minimize the augmented Lagrangian function subject to the original problem's bound constraints. This phase encourages rapid changes to the active set and provides an estimate of the optimal active set. With that active-set estimate, the second phase then solves an equality-constrained QP. (This subproblem gives rise to a KKT system.) A filter method (see, e.g., [36]) is used to dynamically control the accuracy of the bound-constrained solves, thereby eliminating an arbitrary (and sometimes troublesome) sequence of parameters commonly used in augmented Lagrangian techniques.

In the remainder of this section we formally define the QP problem, review its optimality conditions, and give the main assumptions used throughout this paper. In Section 2 we outline the new algorithm and discuss its salient features. In Section 3 we prove global and finite convergence of the algorithm and show that it identifies the optimal active set in a finite number of iterations. Once this active set has been identified, the algorithm may be interpreted as a Newton iteration on the active set (see Section 3.5). In Section 4 we present preliminary numerical results that demonstrate the effectiveness of this approach.

1.1 The Quadratic Program and Its Optimality Conditions

We consider QPs of the form

$$\text{(QP)} \quad \begin{cases} \underset{x \in R^n}{\text{minimize}} & c^T x + \frac{1}{2} x^T H x \\ \text{subject to} & Ax = b, \quad x \geq 0, \end{cases}$$

where b and c are m- and n-vectors, H is a symmetric (and possibly indefinite) $n \times n$ matrix, and A is an $m \times n$ matrix (typically $n \gg m$). QPs with more general upper and lower bounds are easily accommodated by the methods that we discuss. We assume that the constraints are feasible and that A has full row rank. It is straightforward to detect infeasibility by solving the bound-constrained least-squares problem

$$\underset{x}{\text{minimize}} \quad \|Ax - b\|^2 \quad \text{subject to} \quad x \geq 0$$

with a suitable solver. Unless otherwise specified, $\|\cdot\|$ represents the two norm of a vector.

Define the quadratic objective of (QP) by

$$q(x) = c^T x + \tfrac{1}{2} x^T H x$$

and the *Lagrangian* and the *augmented Lagrangian* corresponding to (QP) as

$$\begin{aligned} \mathcal{L}(x, y) &= q(x) - y^T (Ax - b), \\ \mathcal{L}_\rho(x, y) &= \mathcal{L}(x, y) + \tfrac{1}{2} \rho \|Ax - b\|^2, \end{aligned}$$

respectively, where x and the m-vector y are independent variables and $\rho > 0$. When y_k and ρ_k are fixed, we often use the shorthand notation $\mathcal{L}_k(x) \equiv \mathcal{L}_{\rho_k}(x, y_k)$.

Denote the gradient of q by $g(x) = c + Hx$, and define the *first-order multiplier estimate* by

$$\widetilde{y}_\rho(x, y) = y - \rho(Ax - b). \tag{1.1}$$

The first and second derivatives of the augmented Lagrangian $\mathcal{L}_\rho$ with respect to x are

$$\nabla_x \mathcal{L}_\rho(x, y) = g(x) - A^T \widetilde{y}_\rho(x, y), \tag{1.2a}$$

$$\nabla^2_{xx} \mathcal{L}_\rho(x, y) = H + \rho A^T A. \tag{1.2b}$$

We assume that (QP) is feasible and has at least one point (x^*, y^*) that satisfies the first-order KKT conditions.

Definition 1.1 (First-order KKT conditions). *A pair (x^*, y^*) is a first-order KKT point for (QP) if*

$$\min\big(x^*, \nabla_x \mathcal{L}(x^*, y^*)\big) = 0, \tag{1.3a}$$

$$Ax^* = b. \tag{1.3b}$$

The Lagrange multipliers corresponding to the simple bounds ($x \geq 0$) are given by $z^* = \nabla_x \mathcal{L}(x^*, y^*)$. We define the *active* and *inactive* index sets at x as

$$\mathcal{A}(x) = \{j \in 1, \ldots, n \mid [x]_j = 0\} \qquad \text{and} \qquad \mathcal{I}(x) = \{j \in 1, \ldots, n \mid j \notin \mathcal{A}(x)\}.$$

We often use the shorthand notations $\mathcal{A}_k$ and $\mathcal{I}_k$ to indicate the active and inactive sets at an iterate x_k.

Definition 1.2 (Strict complementarity). *The first-order point* (x^*, y^*) *satisfies strict complementarity if* $[\nabla_x \mathcal{L}(x^*, y^*)]_j > 0$ *for all* $j \in \mathcal{A}(x^*)$.

The symbol x^* may denote a (primal) solution of (QP) and may also be used to denote a limit point of the sequence $\{x_k\}$.

A vital component of our algorithm is the concept of a filter [36], which we use to determine the required subproblem optimality and to test acceptance during the linesearch procedure. The filter is defined by a collection of tuples together with a rule that must be enforced among all entries maintained in the filter. We denote the filter at the kth iteration by $\mathcal{F}_k$; it is fully defined in Section 2.1.

1.2 Related Work

Our method is related to a number of nonlinear programming approaches. The two-step aspect of our method is reminiscent of SLP/EQP methods, which have received much attention recently. (For examples of such approaches, see Fletcher and Sainz de la Maza [35] and, more recently, Chin and Fletcher [18] and Byrd et al. [14].) A common theme of these methods is that each solves a "cheaper" linear program (LP) as a subproblem in order to obtain an estimate of the optimal active set; the LP is then followed by the solution of an equality-constrained QP. The idea of using gradient projection to predict the optimal active set has been used in the context of bound-constrained QPs by Moré and Toraldo [56] and by Friedlander and Martínez [38], among others. Bound-constrained QP solvers have also been considered by [5, 19, 20, 25, 27–29].

Our algorithm may be interpreted as a second-order version of the classical augmented Lagrangian algorithm for nonlinear programming (as implemented in LANCELOT [23]); we use the term *bound-constrained Lagrangian* (BCL) for these methods because they involve only bound constraints on each subproblem. BCL methods for general QPs have also recently been considered by Dostál et al. [30] and by Delbos and Gilbert [24]. Other active-set methods for solving QPs include Galahad's QPA [47], BQPD [34], QPOPT [39], and SQOPT [40]. See Gould and Toint [48] for a recent survey on methods for solving QPs.

2 Augmented Lagrangian Filter Algorithm for QPs

Our algorithm differs from the classical bound-constrained augmented Lagrangian (BCL) methods in three important ways: First, the minimization of the augmented Lagrangian provides an estimate of the optimal active set, which is used to define an equality-constrained QP that is solved for a second-order step. This step improves both the reliability and the convergence rate of BCL methods. Second, we use a filter to control various aspects of the

Algorithm 1: Outline of QP Filter Method (QPFIL)

```
initialization: $k \leftarrow 0$, $x_0$ given, initialize $\mathcal{F}_0$
while not optimal do
    1. Approximately minimize $\mathcal{L}_k(x)$ to find an $\widetilde{x}_k$ acceptable to $\mathcal{F}_k$.
    2. Identify an active set $\mathcal{A}_k$ and update the penalty parameter $\rho_{k+1}$.
    3. Update the multiplier estimate: $\widetilde{y}_k \leftarrow y_k - \rho_{k+1}(A\widetilde{x}_k - b)$.
    4. Solve an equality-constrained QP for a second-order step $(\Delta x, \Delta y)$.
    5. Linesearch: find $\alpha$ such that $(\widetilde{x}_k + \alpha\Delta x, \widetilde{y}_k + \alpha\Delta y)$ is acceptable to $\mathcal{F}_k$.
    6. Update the filter $\mathcal{F}_{k+1}$.
    7. $k \leftarrow k + 1$.
end
```

algorithm related to its global convergence properties. The filter allows us to dispense with two forcing sequences commonly used in BCL methods (the subproblem tolerance and the accept/reject threshold for updating the Lagrange multipliers). It also provides a nonmonotone globalization strategy that is more likely to accept steps computed by inexact solves. Third, we exploit the structure of the QP problem to obtain estimates of the required penalty parameter. These estimates are more adaptive than traditional penalty update schemes, which may overestimate the penalty parameter. Algorithm 1 outlines the main steps of our approach.

We emphasize that the main computational components of the algorithm can take advantage of existing matrix-free solvers and are designed to be scalable on high-performance architectures.

At this point, the careful reader may ask why QPFIL uses *both* a filter and a penalty function—after all, filter methods are meant to replace penalty functions (such as the augmented Lagrangian). The reason is simple: We use the augmented Lagrangian to transform the general QP into a bound-constrained QP so that we can use efficient gradient-projection techniques to derive an active-set estimate. We emphasize that global convergence is enforced by the filter and not through the augmented Lagrangian.

A crucial feature of QPFIL is its suitability for high-performance computing. The two computational kernels of the algorithm are the bound-constrained minimization of an augmented Lagrangian function (Step 1) and the solution of an equality-constrained QP (Step 4). Scalable tools that perform well on high-performance architectures exist for both steps. For example, TAO [7] and PETSc [2,3] are suitable, respectively, for the bound-constrained subproblem and the equality-constrained QP. In the remainder of this section we provide details of each step of the QPFIL algorithm.

2.1 An Augmented Lagrangian Filter

The iterations of a BCL method for nonconvex optimization typically are controlled by two fundamental forcing sequences that ensure convergence to a solution. A decreasing sequence $\omega_k \searrow 0$ determines the required optimality of each subproblem solution and controls the convergence of the dual infeasibility. The second decreasing sequence, $\eta_k \searrow 0$, tracks the primal infeasibility $\|Ax - b\|$ and determines whether the penalty parameter ρ_k is increased or left unchanged. Typically the bound constraints are repeated verbatim in the subproblem and enforced at all iterations (see, e.g., [22] for an overview of BCL methods).

In the definition of our filter we use quantities that are analogous to ω_k and η_k. Define

$$\omega(x,y) = \| \min\big(x, \nabla_x \mathcal{L}(x,y)\big) \|, \tag{2.1a}$$
$$\eta(x) = \|Ax - b\|, \tag{2.1b}$$

which are based on the optimality and feasibility of a current pair (x, y) (cf. (1.3)). As we prove in Section 3, such a choice allows us to dispense with the sequences normally found in BCL methods and instead defines these sequences implicitly. We observe that the filter will generally be less conservative than BCL methods in the acceptance of a current subproblem solution or multiplier update. In the remainder of the paper we use the abbreviations

$$\omega_k := \omega(x_k, y_k) \qquad \text{and} \qquad \eta_k := \eta(x_k).$$

Note that $w(x, y)$ defined in (2.1a) is based on the gradient of the Lagrangian function, not on the augmented Lagrangian. Thus, our decision on when to exit the minimization of the current subproblem is based on the optimality of the current subproblem iterate for the original problem, rather than being based on the optimality of the current subproblem as is usually the case in BCL methods. This approach ensures that the subproblem iterations (defined below) always generate solutions that are acceptable to the filter. Another advantage of this definition is that the filter is, in effect, independent of the penalty parameter ρ_k and hence does not need to be updated if ρ_k is increased.

Definition 2.1 (Augmented Lagrangian Filter). *The following rules define an augmented Lagrangian filter:*

1. *A pair (ω_k, η_k) dominates another pair (ω_l, η_l) if $\omega_k \le \omega_l$ and $\eta_k \le \eta_l$ and at least one inequality holds strictly.*

2. *A* filter *$\mathcal{F}$ is a list of pairs (ω_l, η_l) such that no pair dominates another.*

3. *The filter at iteration k is denoted by $\mathcal{F}_k$ and contains pairs of values (ω_l, η_l) from previous iterations.*

4. *We say that (x_{k+1}, y_{k+1}) is* acceptable to the filter *$\mathcal{F}_k$ if and only if*

$$\omega_{k+1} \le \beta\omega_l \quad \textit{or} \quad \eta_{k+1} \le \beta\eta_l - \gamma\omega_{k+1}, \tag{2.2}$$

   for each $(\omega_l, \eta_l) \in \mathcal{F}_k$, where $\beta, \gamma \in (0, 1)$ are constants.

5. *A filter $\mathcal{F}$ contains an entry (the upper bound)*

$$(\bar{\omega}, \bar{\eta}) = (U, 0) \tag{2.3}$$

   where U is a positive constant.

We use the shorthand notation $l \in \mathcal{F}$ to imply that $(\omega_l, \eta_l) \in \mathcal{F}$.

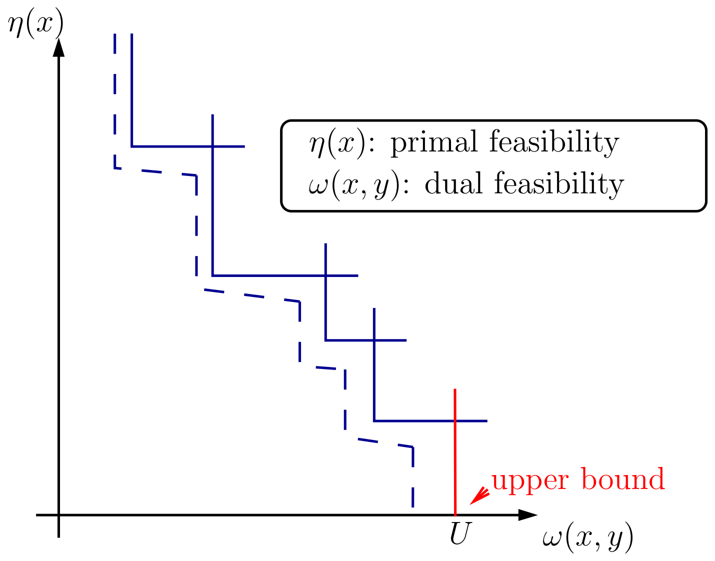
$\eta(x)$
$\eta(x)$: primal feasibility
$\omega(x,y)$: dual feasibility
upper bound
U
$\omega(x,y)$

Figure 1: A typical filter. All pairs (ω, η) that are below and to the left of the envelope (dashed line) are acceptable to the filter (cf. (2.2)).

A typical filter is illustrated in Figure 1. Typical values for the envelope constants are $\beta = 0.999$, $\gamma = 0.001$. A suitable choice for the upper bound U in (2.3) is $U = \delta \max\big(1, \omega(x_0, y_0)\big)$, with $\delta = 1.25$. We note that (2.2) creates a sloping envelope around the filter. Together with (2.3), this implies that a sequence $\{(\omega_k, \eta_k)\}$ of pairs each acceptable to $\mathcal{F}_k$ must satify $\omega_k \to \omega^* = 0$. If the second condition in (2.2) were weakened to $\eta_{k+1} \le \beta\eta_l$, then the sequence of pairs acceptable to $\mathcal{F}_k$ could accumulate to nonstationary points where $\eta_k \to \eta^* = 0$ but $\omega^* > 0$.

A consequence of $\eta(x) \ge 0$ and the sloping envelope is that the upper bound $(U, 0)$ is theoretically unnecessary—the sloping envelope implies an upper bound $U = \eta_{\min}/\gamma$, where $\eta_{\min}$ is the least η_l for all $l \in \mathcal{F}$. In practice, however, we impose the upper bound U in order to avoid generating entries with excessively large values of ω_k.

We remark that the axes in the augmented Lagrangian filter appear to be the reverse of the usual definition (feasibility is on the vertical axis instead of the horizontal axis, as it typically appears in the literature). This reflects the dual view of the augmented Lagrangian: it can be shown that $Ax_k - b$ is a steepest descent direction for the augmented Lagrangian (see, e.g., [11, Section 2.2] or [57, Section 16.6.1]) and that $\omega(x, y)$ is the dual feasibility error. This definition of the filter is similar to the one used in [44]. The gradient of the Lagrangian has also been used in the filter by Ulbrich et al. [63], together with a centrality measure, in the context of interior-point methods.

2.2 Active-Set Prediction and Second-Order Steps

Let $\widetilde{x}_k$ be an approximate minimizer of the augmented Lagrangian $\mathcal{L}_k(x)$ at iteration k. We use this solution to derive an active-set estimate $\mathcal{A}_k := \mathcal{A}(\widetilde{x}_k)$, which in turn is used to define an equality-constrained QP (EQP) in the inactive variables $\mathcal{I}_k := \mathcal{I}(\widetilde{x}_k)$; the variables $\mathcal{A}_k$ are held fixed at the active bounds.

Let c_k and $g_k(x)$ denote the subvectors of c and $g(x)$ corresponding to the indices in $\mathcal{I}_k$. Let H_k be the submatrix formed from the jth rows and columns of H, and let A_k denote the submatrix formed from the jth columns of A, where $j \in \mathcal{I}_k$.

A second-order correction to $\widetilde{x}_k$, given by $\Delta x_{\mathcal{I}} := \widehat{x}_k - \widetilde{x}_k$, may be found by solving the following equality-constrained QP for $\widehat{x}_k$:

$$(\mathrm{EQP}_k) \left\{ \begin{array}{ll} \underset{x}{\text{minimize}} & c^T x + \frac{1}{2} x^T H x \\ \text{subject to} & Ax = b \\ & x_{\mathcal{A}_k} = [\widetilde{x}_k]_{\mathcal{A}_k}. \end{array} \right. \tag{2.4}$$

Note that in the formulation (QP), $[\widetilde{x}_k]_{\mathcal{A}_k} = 0$. We use this formulation to emphasize that QPs with more general bounds would have EQP subproblems with fixed variables that may be nonzero. The first-order conditions for (2.4) are used to generate a search direction from the current point $(\widetilde{x}_k, \widetilde{y}_k)$:

$$\begin{pmatrix} -H_k & A_k^T \\ A_k & \end{pmatrix} \begin{pmatrix} \Delta x_{\mathcal{I}} \\ \Delta y \end{pmatrix} = \begin{pmatrix} g_k(\widetilde{x}_k) - A_k^T \widetilde{y}_k \\ b - A\widetilde{x}_k \end{pmatrix}. \tag{2.5}$$

A projected search in the full space is then based on the vector $(\Delta x, \Delta y)$, where

$$\Delta x = (\Delta x_{\mathcal{A}}, \Delta x_{\mathcal{I}}) \quad \text{and} \quad \Delta x_{\mathcal{A}} \equiv 0.$$

Note that Step 1 of Algorithm 1 requires that the approximate augmented Lagrangian minimizer $\widetilde{x}_k$ be acceptable to the filter. Moreover, as we demonstrate in Section 3.1, the first-order multiplier estimate $\widetilde{y}_k$ must also be acceptable to the filter. These two properties ensure that even if a linesearch along $(\Delta x, \Delta y)$ fails to obtain a positive steplength α such that $(\widetilde{x}_k + \alpha\Delta x, \widetilde{y}_k + \alpha\Delta y)$ is acceptable to the filter, the algorithm can still make progress with the first-order step alone. In this case, $\alpha = 0$, and the algorithm relies on the progress of the standard BCL iterations.

2.3 Estimating the Penalty Parameter

It is well known that BCL methods, under standard assumptions, converge for all large-enough values of the penalty parameter ρ_k. The threshold value $\bar{\rho}$ is never computed explicitly; instead, BCL methods attempt to "discover" the threshold value by increasing ρ_k in stages. Typically the norm of the constraint violation is used to guide the decisions on when to increase the penalty parameter: a linear decrease (as anticipated by the BCL local convergence theory) signals that the penalty parameter may be held constant; less than linear convergence—or a large increase in constraint violations—indicates that a larger ρ_k is needed.

When the constraints are nonlinear, the penalty-parameter threshold and the initial Lagrange multiplier estimates are closely coupled. Poor estimates y_k of y^* imply that a larger ρ_k is needed to induce convergence. This coupling is fully described by Bertsekas [10, Proposition 2.4]. When the constraints are linear, however, the Lagrange multipliers do not appear in (1.2b), and we see that y_k and ρ_k are essentially decoupled—the curvature of $\mathcal{L}_k$ can be influenced by changing ρ_k alone. This observation is illustrated in Figure 2,

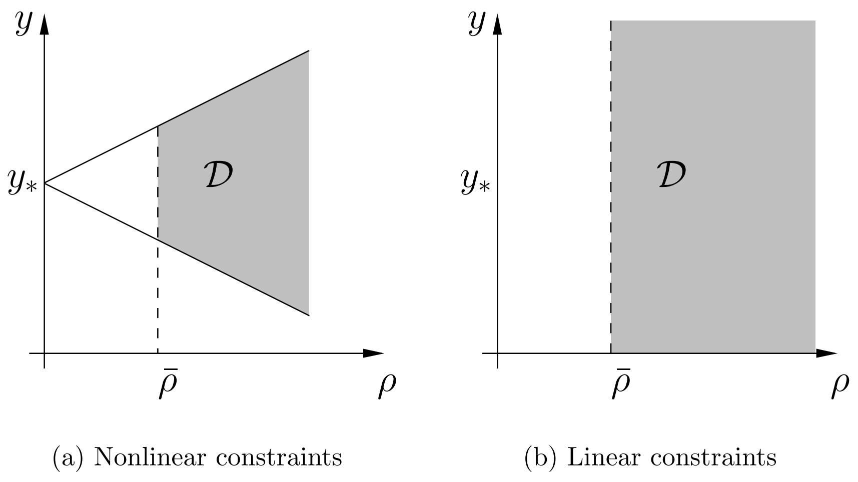

Figure 2: The sets $\mathcal{D}$ illustrate the required penalty parameter for the BCL method when the constraints are either nonlinear or linear.

in which the left figure corresponds to nonlinear constraints and the right figure to linear constraints. The regions in the penalty/multiplier plane for which BCL methods converge are indicated by the shaded regions $\mathcal{D}$. The result below provides an explicit threshold value $\bar{\rho}$ needed ensure that the Hessian of the augmented Lagrangian is positive definite. (A positive multiple of this quantity is enough to induce convergence.) For the remainder of this section only, let $\lambda_{\min}(\cdot)$ and $\sigma_{\min}(\cdot)$, respectively, denote the leftmost eigenvalue and the smallest singular value of a matrix.

Lemma 2.2. *Suppose that $p^THp > 0$ for all $p \neq 0$ such that $Ap = 0$ and A has full row rank. Then $H + \rho A^TA$ is positive definite if and only if*

$$\rho > \bar{\rho}_{\min} := \lambda_{\min}\big(A(H + \gamma A^TA)^{-1}A^T\big)^{-1} - \gamma I, \tag{2.6}$$

for any $\gamma \geq 0$ such that $H + \gamma A^TA$ is nonsingular.

Proof. The required result follows from Bertsekas [10, Proposition 2.5], where the Jacobian and Hessian are taken as constant. $\square$

The bound provided by Lemma 2.2 is sharp: it is both necessary and sufficient. However, the formula on the right-hand side of (2.6) is unsuitable for large-scale computation. The following lemma develops a lower bound for the required ρ that is more easily computable.

Lemma 2.3. *Under the conditions of Lemma 2.2,*

$$\frac{\max\{0, -\lambda_{\min}(H)\}}{\sigma_{\min}(A)^2} > \bar{\rho}_{\min}. \tag{2.7}$$

Proof. Consider unit-norm vectors p such that $Ap \neq 0$. (Otherwise, $p^T(H + \rho A^TA)p = p^THp > 0$ for all p such that $Ap = 0$.) Let $U = [u_1 \cdots u_m]$ be the orthogonal left-singular vectors of A, and let $\Sigma = \operatorname{diag}(\sigma_i)$ be the singular values, with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_m \equiv \sigma_{\min} > 0$ (we assume that A has full rank). Then p can be expressed as

$$p = \sum_{i=1}^{m} \alpha_i u_i \quad \text{with} \quad \sum_{i=1}^{m} \alpha_i^2 = 1,$$

for some scalars α_i not all zero. Then $A^TA = U\Sigma^T\Sigma U^T$, and

$$p^TA^TAp = \left(\sum_{i=1}^{m} \alpha_i u_i^T\right)(U\Sigma^T\Sigma U^T)\left(\sum_{i=1}^{m} \alpha_i u_i\right) = \sum_{i=1}^{m} \alpha_i^2 \sigma_i^2. \tag{2.8}$$

Similarly, let $Q = [q_1 \cdots q_n]$ be the orthogonal eigenvectors of H, and let $\Lambda = \operatorname{diag}(\lambda_i)$ be the eigenvalues, with $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \equiv \lambda_{\min}$. Then there exist scalars β_i not all zero such that

$$p = \sum_{i=1}^{n} \beta_i q_i \quad \text{with} \quad \sum_{i=1}^{n} \beta_i^2 = 1$$

and

$$p^THp = \left(\sum_{i=1}^{n} \beta_i q_i^T\right) H \left(\sum_{i=1}^{n} \beta_i q_i\right) = \sum_{i=1}^{n} \beta_i^2 \lambda_i. \tag{2.9}$$

Therefore, (2.8) and (2.9) imply that

$$\begin{aligned} p^T(H + \rho A^TA)p &= \sum_{i=1}^{n} \beta_i^2 \lambda_i + \rho \sum_{i=1}^{m} \alpha_i^2 \sigma_i^2 \\ &> \min(0, \lambda_{\min}) \sum_{i=1}^{n} \beta_i^2 + \rho\sigma_{\min}^2 \sum_{i=1}^{m} \alpha_i^2 \\ &= \min(0, \lambda_{\min}) + \rho\sigma_{\min}^2, \end{aligned}$$

and so $H + \rho A^TA$ is positive definite if $\min(0, \lambda_{\min}) + \rho\sigma_{\min}^2 > 0$ or, equivalently, if

$$\rho > \frac{\max(0, -\lambda_{\min})}{\sigma_{\min}^2}. \tag{2.10}$$

Because the bound $\bar{\rho}_{\min}$ in Lemma 2.2 is sharp, (2.10) implies that the required bound (2.7) holds. □

For a given active set $\mathcal{A}_k$, Lemma 2.3 implies that

$$\rho_k > \frac{\max\{0, -\lambda_{\min}(H_k)\}}{\sigma_{\min}(A_k)}$$

is sufficient to ensure that $\mathcal{L}_k$ is convex on that subspace. Note that this lower bound tends to infinity as the smallest singular value of A_k tends to zero. This property is consistent with (2.6), where we see that if A_k is rank deficient, then the required bound in Lemma 2.2 does not exist. In Section 3 we show that for a given optimal active set, a multiple of this bound is required to induce convergence to an optimal solution in our method.

We are not entirely satisfied with Lemma 2.3 because it requires an estimate (or at least a lower bound) of the smallest singular value of the current A_k. This calculation can be expensive. One possibility for estimating this value is to use a Lanczos bidiagonalization procedure, as implemented in PROPACK [50].

Ideally, we would compute the penalty value according to any of these rules. However, this approach would be prohibitive in terms of computational effort for the size of problems of interest. A simple approximation that we have used instead is

$$\bar{\rho}(\mathcal{A}_k) = \max\left\{1,\ \frac{\|H_k\|_1}{\max\left\{\frac{1}{\sqrt{|\mathcal{I}_k|}}\|A_k\|_\infty\ ,\ \frac{1}{\sqrt{m}}\|A_k\|_1\right\}}\right\},$$

where $|\mathcal{I}_k|$ is the number of free variables and m is the number of general equality constraints. We note that we do not require an accurate estimate of the penalty parameter because we are using $2\bar{\rho}$ in the algorithm and our filter does not contain the penalty parameter. Thus we expect our method to be quite insensitive to poor estimates of the penalty parameter.

2.4 Inner Minimization of the Augmented Lagrangian

Like classical BCL methods, our method approximately minimizes a sequence of bound-constrained subproblems

$$\underset{x}{\text{minimize}} \quad \mathcal{L}_k(x) \quad \text{subject to} \quad x \geq 0, \tag{2.11}$$

where $\mathcal{L}_k(x) := \mathcal{L}_{\rho_k}(x, y_k)$ and the multiplier estimate y_k and penalty parameter ρ_k are held fixed. Instead of optimizing (2.11) to a prescribed tolerance, however, each iteration of the inner algorithm approximately optimizes (2.11) in stages (i.e., a few iterations of some minimization procedure are applied); the current iterate and the implied first-order multiplier (1.1) are tested for acceptability against the current filter. The penalty parameter is also updated at each iteration to ensure that it satisfies the bound implied by Lemma 2.2. In practice, we may also wish to increase ρ if the optimality measure approaches zero much faster than feasibility. The inner minimization terminates when the current iterates are acceptable to the filter and the penalty parameter is large enough for the current active set.

The inner-minimization algorithm is described in Algorithm 2. A superscript j indicates an iteration of the inner minimization.

In classical BCL methods, the gradient of the augmented Lagrangian at the current iterate x^j and multiplier estimate $\bar{y}$ is used to test termination of the inner iterations. In contrast, the test in Step 24 of Algorithm 2 is based on the norm of the (usual) Lagrangian function at x^j and the implied first-order multiplier estimate $\bar{y} - \rho^j(Ax^j - b)$. We note that because

$$\nabla_x \mathcal{L}_\rho(x^j, \bar{y}) = \nabla_x \mathcal{L}(x^j, y^j),$$

these criteria are in fact identical. Algorithm 2 adds the current primal infeasibility η^j as an additional criterion.

To establish that our algorithm finitely identifies the optimal active set, we need to be more specific as to what we mean by "approximately minimizing" the augmented Lagrangian

Algorithm 2: Bound-Constrained Lagrangian Filter (BCLFIL)

```
Input: $x^0$, $\bar{y}$, $\rho^0$, $\mathcal{F}$
Output: $\widetilde{x}$, $\widetilde{y}$, $\widetilde{\rho}$, $\widetilde{\omega}$, $\widetilde{\eta}$
1  $j \leftarrow 0$
2  converged $\leftarrow$ false
3  repeat
5      Find a point $x^{j+1}$ that approximately minimizes $\mathcal{L}_{\rho^j}(x, \bar{y})$ subject to $x \geq 0$.
7      $\mathcal{A}^{j+1} \leftarrow \mathcal{A}(x^{j+1})$                [update active set]
9      $\bar{\rho}^{j+1} \leftarrow \bar{\rho}(A^{j+1})$                [compute required penalty parameter]
11     if $\rho^j < \bar{\rho}^{j+1}$ then
13         $\rho^{j+1} \leftarrow 2\bar{\rho}^{j+1}$                [increase penalty parameter]
14     else
16         $y^{j+1} \leftarrow \bar{y} - \bar{\rho}^j(Ax^{j+1} - b)$                [provisional multiplier update]
18         $\omega^{j+1} \leftarrow \omega(x^{j+1}, y^{j+1})$                [update dual infeasibility]
20         $\eta^{j+1} \leftarrow \eta(x^{j+1})$                [update primal infeasibility]
22         $\rho^{j+1} \leftarrow \rho^j$                [keep penalty parameter]
24         if $(\omega^{j+1}, \eta^{j+1})$ is acceptable to $\mathcal{F}$ then converged $\leftarrow$ true
25     end
27     $j \leftarrow j + 1$
28 until converged
29 $\widetilde{x} \leftarrow x^j$, $\widetilde{y} \leftarrow y^j$, $\widetilde{\rho} \leftarrow \rho^j$, $\widetilde{\omega} \leftarrow \omega^j$, $\widetilde{\eta} \leftarrow \eta^j$
```

in Step 5 of Algorithm 2. In particular, we assume that this approximate minimization reduces the objective by at least as much as does a Cauchy point of a projected-gradient method (see, e.g., [59, Section 16.6]). This is a mild assumption that is satisfied by most globally convergent bound-constrained solvers. In practice, we perform one or two steps of a bound-constrained optimization algorithm and then test the multiplier and the filter acceptance. This requirement is often weaker than traditional augmented Lagrangian methods, which must reduce the projected gradient beyond a specified tolerance ω_k.

2.5 Detailed Algorithm Statement

Our algorithm is structured around *outer* and *inner* iterations. The outer iterations handle management of the filter, the solution of (EQP$_k$), and the subsequent linesearch. The inner iterations minimize the augmented Lagrangian function, update the multipliers and the penalty parameter, and identify a candidate set of active constraints used to define (EQP$_k$) for the outer iteration. Thus, the inner iteration performs Steps 1–3 of Algorithm 1.

Algorithm 3 does not explicitly have the feasibility restoration phase needed by other filter methods. The reason is that the approximate minimization of the augmented Lagrangian corresponds to the minimization of the feasibility measure (recall that the axes in our filter definition are reversed). In addition, we can also interpret the second-order step as a (primal) feasibility restoration because any solution of the EQP provides a feasible point of the QP (with $\eta(x) = 0$). It may be useful to add a restoration phase to the algorithm, however, in order to discover whether the problem is infeasible or whether η_k grows too large relative to

Algorithm 3: QP Filter Method (QPFIL)

```
    Input: $x_0$, $y_0$
    Output: $x^*$, $y^*$
 1  Set penalty parameter $\rho_0 > 0$.
 2  Set positive filter envelope parameters $\beta, \gamma < 1$.
 3  Set filter upper bound $U \leftarrow \gamma \max(1, \|Ax_0 - b\|)$.
 4  Add $(U, 0)$ to filter $\mathcal{F}_0$.
 5  Set minimum steplength $\alpha_{\min} > 0$.
 6  Compute infeasibilities $\omega_0 \leftarrow \omega(x_0, y_0)$ and $\eta_0 \leftarrow \eta(x_0)$.
 7  $k \leftarrow 0$
 9  if $\omega_0 > 0$ and $\eta_0 > 0$ then add $(\omega_0, \eta_0)$ to $\mathcal{F}_0$.
10  while not optimal do
12      $(\widetilde{x}_k, \widetilde{y}_k, \rho_{k+1}, \omega_{k+1}, \eta_{k+1}) \leftarrow$ BCLFIL$(x_k, y_k, \rho_k, \mathcal{F}_k)$
14      $\mathcal{A}_k \leftarrow \mathcal{A}(\widetilde{x}_k)$
16      Find $(\Delta x_k, \Delta y_k)$ that solves (2.5)
18      Find $\alpha_k \in [\alpha_{\min}, 1]$ such that $(\widetilde{x}_k + \alpha\Delta x_k, \widetilde{y}_k + \alpha\Delta y_k)$ is acceptable to $\mathcal{F}_k$
20      if linesearch failed then $\alpha_k = 0$
22      $(x_{k+1}, y_{k+1}) \leftarrow (\widetilde{x}_k + \alpha_k\Delta x_k, \widetilde{y}_k + \alpha\Delta y_k)$
24      if $\alpha_k > 0$ then $\omega_{k+1} \leftarrow \omega(x_{k+1}, y_{k+1})$ and $\eta_{k+1} \leftarrow \eta(x_{k+1})$
26      if $\omega_{k+1} > 0$ then
28          $\mathcal{F}_{k+1} \leftarrow \mathcal{F}_k \cup \{(\omega_{k+1}, \eta_{k+1})\}$
30          Remove redundant entries from $\mathcal{F}_{k+1}$
32          if $\eta_{k+1} = 0$ then update upper bound $U$
33      end
35      $k \leftarrow k + 1$
36  end
37  $x^* \leftarrow x_k$, $y^* \leftarrow y_k$
```

ω_k. We leave this option for a later implementation.

In Step 18 of Algorithm 3 we perform a filter linesearch by trying a sequence of steps

$$\alpha = \gamma^k, \qquad k = 0, 1, 2, \ldots$$

for some constant $\gamma \in (0, 1)$ until an acceptable point is found or until $\alpha < \alpha_{\min}$, where $\alpha_{\min} > 0$ is a constant parameter. The parameter $\alpha_{\min}$ is needed because the first-order point $(\widetilde{x}_k, \widetilde{y}_k)$ could lie in a corner of the filter with the second-order step pointing into the filter. In that case there exists no $\alpha > 0$ that yields an acceptable step. Other choices for deciding when to terminate the linesearch are possible, based, for example, on requiring that the new filter area induced by the linesearch step be larger than the new filter area induced by the first-order step.

The filter update in Step 28 of Algorithm 3 removes redundant entries that are dominated by a new entry. The upper bound $(U, 0)$ also allows us to manage the number of filter entries that we wish to store. If this number is exceeded, then we can reset the upper bound as $U = \max_i\{\omega_i \mid \omega_i \in \mathcal{F}_k\}$ and subsequently delete dominated entries from $\mathcal{F}_k$, thus reducing the number of filter entries.

3 Global Convergence of Algorithm QPFIL

Global convergence of QPFIL is based on progress of the inner iterations. The second-order updates in Steps 16–22 serve only to accelerate convergence. Therefore, we establish global convergence of QPFIL by analyzing a first-order version of Algorithm 3 that does not use the second-order updates. We also establish that this first-order version of the algorithm identifies the optimal active set in a finite number of iterations. The second-order corrections in Steps 16–22 consequently give an exact solution to (QP), and the entire algorithm is therefore finitely convergent.

3.1 Preliminaries

The following assumptions hold implicitly throughout.

Assumption 3.1.

[A1] The sequence $\{x_k\}$ lies in a compact set. Thus, there exist at least one limit point x^* and a corresponding convergent subsequence $\mathcal{K}$.

[A2] The reduced Jacobian $A_* := A(:, \mathcal{A}^*)$ has full rank at every limit point x^*.

Next, we show that the penalty parameter eventually settles down and is not changed for k sufficiently large.

Lemma 3.2. *The penalty parameter is updated finitely often.*

Proof. This follows from the fact that there exist only a finite number of different active sets that could result in a penalty-parameter update. □

The following lemma demonstrates that the algorithm is well defined, by showing that the inner iteration always terminates after a finite number of iterations with a point acceptable to the current filter.

Lemma 3.3. *The inner iteration is finite.*

Proof. Steps 9 and 26 of Algorithm 3 guarantee that $\omega_j > 0$ for all $j \in \mathcal{F}_k$. Moreover, Algorithm 2 generates iterates $\omega^j \to 0$. Thus, the inner iterations find a point acceptable to $\mathcal{F}_k$ within a finite number of iterations. □

Next, we show that every limit point of the primal-dual pair (x_k, y_k) satisfies (1.3a) and is thus dual feasible.

Lemma 3.4. *Any limit point (x^*, y^*) of $\{(x_k, y_k)\}$ is dual feasible, namely, $\omega_k \equiv \omega(x_k, y_k) \to 0$.*

Proof. We consider two mutually exclusive cases, depending on whether a finite or an infinite number of entries are added to the filter. If a finite number of entries are added to the filter (i.e., if Step 26 tests true only finitely many times), then it follows that $\omega_k = 0$ for all k sufficiently large. The required result then follows immediately. If, on the other hand, an infinite number of entries (ω_k, η_k) are added to the filter, then the required result follows from [18, Lemma 1] (with $f(x) = \eta(x)$ and $h(x) = \omega(x, y)$), because $\eta(x)$ is trivially bounded below. □

3.2 Least-Squares Multiplier Estimates

Define the *least-squares multiplier estimate* $\widehat{y}(x)$ as the unique solution of the least-squares problem

$$\widehat{y}(x) \equiv \arg\min_{y} \|[g(x)]_{\mathcal{I}^*} - A_*^T y\|. \tag{3.1}$$

Because the least-squares multiplier estimate is unique, there exists a positive constant α_1 such that

$$\|\widehat{y}(\widetilde{x}_k) - \widehat{y}(x^*)\| \le \alpha_1 \|\widetilde{x}_k - x^*\| \tag{3.2}$$

for all k sufficiently large. Note that the definition of $\widehat{y}$ requires a priori knowledge of the bounds that are active at the solution; however, $\widehat{y}$ is used only for analysis and is never computed.

Lemma 3.5. *Suppose that y is an approximate least-squares solution of* (3.1) *for some x. Then there exists some positive constant α_2 such that*

$$\|\widehat{y}(x) - y\| \le \alpha_2 \|[g(x)]_{\mathcal{I}^*} - A_*^T y\|. \tag{3.3}$$

Proof. Let $\widehat{r}(x)$ and r be the least-squares residuals associated with $\widehat{y}(x)$ and y, respectively, so that $A_*^T\widehat{y}(x) + \widehat{r}(x) = b$ and $A_*^T y + r = b$. Then

$$A_*^T(\widehat{y}(x) - y) + \widehat{r}(x) - r = 0,$$

and because $A_* r(x) = 0$, it follows that

$$A_* A_*^T(\widehat{y}(x) - y) + A_*\widehat{r}(x) = 0.$$

Because A_* has full rank, it is straightforward to show that there exists a positive constant α_2 such that $\|\widehat{y}(x) - y\| \le \alpha_2\|\widehat{r}(x)\|$, and the required result follows immediately from the definition of $\widehat{r}(x)$ □

3.3 Convergence of First-Order Algorithm

For this section only, we consider a simplified algorithm that skips the second-order update (Steps 16–24 of Algorithm 3). We prove that the first-order sequence $\{(\widetilde{x}_k, \widetilde{y}_k)\}$ generated in Step 12 converges to a stationary point of (QP). This result is of interest also within the context of more established BCL methods because it illustrates how a filter can be used in place of the two arbitrary forcing sequences (ω_k and η_k) commonly associated with augmented Lagrangian methods.

We refer to the sequence $\{(\widetilde{x}_k, \widetilde{y}_k)\}$ coming from the minimization of the augmented Lagrangian function as a sequence of *Cauchy points*; the intention is to emphasize that these solutions can be interpreted as steepest-ascent steps of the augmented Lagrangian function and thus can yield only linear convergence.

Theorem 3.6. *Consider a version of Algorithm 3 that skips Steps 16–24. Assume that the algorithm generates a sequence of Cauchy points $\{(\widetilde{x}_k, \widetilde{y}_k)\}$ and that x^* is the* single *limit point of $\{\widetilde{x}_k\}$. Then $\widetilde{y}_k \to y^*$, where $y^* \equiv \widehat{y}(x^*)$, and (x^*, y^*) is a KKT point of (QP).*

Proof. Assume that the algorithm does not terminate finitely. Let

$$\widetilde{z}_k := \nabla_x \mathcal{L}(\widetilde{x}_k, \widetilde{y}_k) \equiv g(\widetilde{x}_k) - A_*^T \widetilde{y}_k.$$

By Lemma 3.4,

$$\omega_k \equiv \| \min(\widetilde{x}_k, \widetilde{z}_k)\| \to 0. \tag{3.4}$$

Now consider the inactive components $\mathcal{I}^*$ of x^*, and observe that for some positive constant τ, $[x^*]_i \geq \tau > 0$ for all $i \in \mathcal{I}^*$. Thus, it follows that, for k sufficiently large, $[x_k]_i \geq \tau/2 > 0$ for all $i \in \mathcal{I}^*$. Because $\omega_k \to 0$, $\min([\widetilde{x}_k]_i, [\widetilde{z}_k]_i) = [\widetilde{z}_k]_i$, so

$$\|[\widetilde{z}_k]_{\mathcal{I}^*}\| \leq \| \min(\widetilde{x}_k, \widetilde{z}_k)\| \equiv \omega_k. \tag{3.5}$$

From the triangle inequality,

$$\|\widetilde{y}_k - y^*\| \leq \|\widehat{y}(\widetilde{x}_k) - \widetilde{y}_k\| + \|\widehat{y}(\widetilde{x}_k) - y^*\|. \tag{3.6}$$

Using inequality (3.5) in (3.3), with $x = \widetilde{x}_k$ and $y = \widetilde{y}_k$, we deduce that

$$\|\widehat{y}(\widetilde{x}_k) - \widetilde{y}_k\| \leq \alpha_1 \omega_k. \tag{3.7}$$

Substituting (3.2) and (3.7) into (3.6), we obtain

$$\|\widetilde{y}_k - y^*\| \leq \alpha_1 \omega_k + \alpha_2 \|\widetilde{x}_k - x^*\|,$$

and so $\widetilde{y}_k \to y^*$, as required (recall that $y^* := \widehat{y}(x^*)$).

Now note that $\widetilde{z}_k \to z^* \equiv \nabla_x \mathcal{L}(x^*, y^*)$, so (3.4) implies that $\min(x^*, z^*) = 0$. Therefore, (x^*, y^*) satisfies (1.3a). The "single limit point" assumption on $\widetilde{x}_k$ and the first-order multiplier update $\widetilde{y}_k = \widetilde{y}_{k-1} + \rho_k(A\widetilde{x}_k - b)$ (see Step 16 of Algorithm 2) imply that $\|\widetilde{y}_k - \widetilde{y}_{k-1}\| \to 0$. By Lemma 3.2, ρ_k is bounded for all k, so

$$\|\widetilde{y}_k - \widetilde{y}_{k-1}\| = \rho_k \|A\widetilde{x}_k - b\| \to 0, \tag{3.8}$$

and x^* satisfies (1.3b). Hence, (x^*, y^*) is a KKT pair of (QP). $\square$

The single-limit-point assumption of Theorem 3.6 excludes the unacceptable situation in which consecutive minimizations of the augmented Lagrangian converge to different local minimizers. In that case, the corresponding Lagrange multiplier updates would not have a limit point, and the conclusion of (3.8) would fail to hold. A similar assumption is made implicitly in a classical proof of convergence of the augmented Lagrangian method: Bertsekas [10, Proposition 2.n] assumes that all minimimizers of the augmented Lagrangian fall within the same neighborhood.

3.4 Finite Identification of the Active Set and Global Convergence

An interesting feature of the QPFIL algorithm is its finite identification of the optimal active set: the gradient of the augmented Lagrangian "reveals" the optimal active set after a finite number of iterations. This key property is based on the requirement that a gradient projection step on the augmented Lagrangian subproblem must ensure at least a Cauchy decrease. This is sufficient to identify the correct active constraints.

Theorem 3.7. *Assume that the inner minimization performs a gradient projection that ensures at least Cauchy decrease on the augmented Lagrangian, that (QP) satisfies strict complementarity (Definition 1.2), and that $(x_k, y_k) \to (x^*, y^*)$, which is a local minimizer of (QP). Then Algorithm 3 identifies the correct active set in a finite number of iterations.*

Proof. As a consequence of the convergence of (x_k, y_k) and strict complementarity, it follows that there exists a neighborhood $\mathcal{N}_\epsilon$ of (x^*, y^*) such that for all $(x_k, y_k) \in \mathcal{N}_\epsilon$ there exist constants $c_1, c_2, \tau > 0$ (with $\tau \gg \epsilon$), independent of k and ϵ, such that

$$\|Ax_k - b\| \le c_1\epsilon \tag{3.9a}$$
$$\left[c + Hx_k - A^T y_k\right]_i \ge \tau \quad \forall i \in \mathcal{A}_* \tag{3.9b}$$
$$[x_k]_i \le \epsilon \quad \forall i \in \mathcal{A}_* \tag{3.9c}$$
$$\left| \left[c + Hx_k - A^T y_k\right]_i \right| \le c_2\epsilon \quad \forall i \in \mathcal{I}_* \tag{3.9d}$$
$$[x_k]_i \ge \tau \quad \forall i \in \mathcal{I}_*. \tag{3.9e}$$

The gradient projection in the inner iteration computes the projected gradient path and then finds the first minimum of the augmented Lagrangian along this piecewise linear path. The proof largely follows the derivation of the breakpoints and local minima presented in [59, Section 16.6].

We now consider the projected gradient path for the augmented Lagrangian. Because the penalty parameter is updated finitely often (see Lemma 3.2), we can assume that the penalty parameter $\rho_k \equiv \rho$ is fixed. It follows from (3.9) that there exists a constant $c_3 > 0$ such that

$$[\nabla\mathcal{L}_k]_i \ge c_3\tau, \quad \forall i \in \mathcal{A}_* \qquad \text{and} \qquad |[\nabla\mathcal{L}_k]_i| \le c_2\epsilon, \quad \forall i \in \mathcal{I}_*. \tag{3.10}$$

The breakpoints along the piecewise linear projected gradient path from x_k in the direction $-\nabla\mathcal{L}_k$ are given by

$$\bar{t}_i = \begin{cases} [x_k]_i/[\nabla\mathcal{L}_k]_i & \text{if } \nabla\mathcal{L}_k > 0, \\ \infty & \text{otherwise.} \end{cases}$$

Observe that (3.10), (3.9c), and (3.9e) together imply that the breakpoints $\bar{t}_i$ satisfy

$$\bar{t}_i \le \frac{\epsilon}{c_3\tau}, \quad \forall i \in \mathcal{A}_* \qquad \text{and} \qquad \bar{t}_i \ge \frac{\tau}{c_2\epsilon}, \quad \forall i \in \mathcal{I}_*. \tag{3.11}$$

Because $\tau \gg \epsilon$, it follows that the breakpoints corresponding to active constraints $i \in \mathcal{A}_*$ are much smaller than the breakpoints corresponding to inactive constraints $i \in \mathcal{I}_*$. Note also that if $[x_k]_i = 0$, then $\bar{t}_i = 0$. This bound therefore remains active because $[\nabla\mathcal{L}_k(x_k)]_i \ge \tau > 0$.

The piecewise linear projected gradient path can now be parameterized in $t \ge 0$:

$$x_i(t) = \begin{cases} [x_k]_i - t[\nabla\mathcal{L}_k]_i & \text{if } t \le \bar{t}_i, \\ [x_k]_i - \bar{t}_i[\nabla\mathcal{L}_k]_i & \text{otherwise.} \end{cases}$$

Next, we remove duplicate and zero entries from the breakpoints $\{\bar{t}_1, \dots, \bar{t}_n\}$ and sort the remaining entries into an ordered sequence

$$0 < t_1 < t_2 < \cdots < t_{a-1} < t_a < t_i < t_{i+1} \dots .$$

Observe that (3.11) implies that this ordering separates the active indices $1, \dots, a$ from the inactive indices $i, i+1, \dots$, and that

$$t_a \le \frac{\epsilon}{c_3 \tau} \ll \frac{\tau}{c_2 \epsilon} \le t_i.$$

We now show that the first minimizer of the augmented Lagrangian occurs in the interval $[t_a, t_i]$ for ϵ sufficiently small, and therefore the correct active set is identified. We must first demonstrate that the augmented Lagrangian has no minimizer in any of the intervals $[t_{j-1}, t_j]$ for $j \le a$. Let $j \le a$, and consider the piecewise search direction on $[t_{j-1}, t_j]$:

$$p_i^{j-1} = \begin{cases} [-\nabla \mathcal{L}_k(x_k)]_i & \text{if } t_{j-1} \le \overline{t_i}, \\ 0 & \text{otherwise.} \end{cases} \tag{3.12}$$

Next, consider the path segment given by

$$x(t) = x(t_{j-1}) + \Delta t \, p^{j-1} \quad \text{for} \quad \Delta t \in [0, t_j - t_{j-1}],$$

and look for a minimizer of the augmented Lagrangian in this segment. Expanding the Lagrangian along this segment, we compute the directional gradient on $[t_{j-1}, t_j]$ as

$$-f'_{j-1} = -\nabla \mathcal{L}_k^T p^{j-1} - x(t_{j-1})^T (H + \rho A^T A) p^{j-1}.$$

From (3.10) and (3.12), it follows that the first term on the right-hand side above is bounded below by $c_3^2 \tau^2$, while the second term is $\mathcal{O}(\epsilon \tau)$. Therefore, the first term dominates, and $-f'_{j-1} \ge c_3^2 \tau^2$ for $j \le a+1$.

Next, observe that the directional Hessian on $[t_{j-1}, t_j]$,

$$f''_{j-1} = p^{j-1}(H + \rho A^T A) p^{j-1},$$

is bounded because $(H + \rho A^T A)$ and p^{j-1} are bounded. Using (3.10) and (3.12), we conclude that there exist constants $c_4 > 0$ and $\kappa > 0$, independent of ϵ, such that

$$c_4 \tau^2 \le f''_{j-1} \le \kappa.$$

Because $f''_{j-1} > 0$, the minimizer of the augmented Lagrangian on the segment $[t_{j-1}, t_j]$ is given by

$$t_j^* = \min \left\{ \frac{-f'_{j-1}}{f''_{j-1}}, t_j - t_{j-1} \right\}.$$

The first term in the minimum can be bounded below by $c_4 \tau^2 / \kappa \gg \epsilon$, and the second term is $\mathcal{O}(\epsilon)$. Thus, for ϵ sufficiently small, the minimum of the quadratic on the segment $[t_{j-1}, t_j]$ occurs at t_j for all $j \le a$, and the projected gradient search proceeds to the next segment $[t_j, t_{j+1}]$. Repeating this argument for all $j \le a$ shows that there is no minimum of the augmented Lagrangian in any of the segments $[t_{j-1}, t_j]$ for $j \le a$. Hence, all active constraints are identified correctly.

Next, we show that the interval $[t_a, t_i]$ contains a minimum of the augmented Lagrangian in its interior. We can use the same estimates as above for the directional gradient and

Hessian because the left-hand boundary corresponds to an active index. Thus, we again consider the case where $f_a'' = \mathcal{O}(\kappa) > 0$ and $f_a' = \mathcal{O}(\tau^2) > 0$. It follows that the quotient $-f_a/f_a''$ is a constant independent of ϵ. However, the right-hand boundary of the segment

$$\Delta t \in [0, t_i - t_a] = [0, \tau\mathcal{O}(\epsilon^{-1})]$$

becomes large as ϵ becomes small. Thus, for ϵ sufficiently small, the minimum of the augmented Lagrangian occurs in the interior of the segment $[t_a, t_i]$, and no additional inactive constraints are identified as active. $\square$

Theorem 3.7 implies the following corollary, which establishes global convergence of Algorithm 3.

Corollary 3.8. *Let the assumptions of Proposition 3.7 hold, and assume in addition that the augmented system (2.5) in Step 4 of Algorithm 3 is solved exactly. Then the algorithm terminates finitely at a KKT point of (QP).*

Proof. The proof follows because QPFIL identifies the correct active set in a finite number of iterations and an exact solve of (2.5) subsequently gives the solution of (QP). $\square$

3.5 Equivalence of the Second-Order Step and Newton's Method

Once the correct active set identified, we may interpret the second-order step (2.5) as a Newton step on the stationary conditions of the augmented Lagrangian on the set of variables that are inactive. Consider $\nabla_{xy}\mathcal{L}_\rho(x,y) = 0$ in the subspace of inactive (primal) variables:

$$\begin{pmatrix} [\nabla_x \mathcal{L}_{\rho_k}(x,y)]_{\mathcal{I}_k} \\ \nabla_y \mathcal{L}_{\rho_k}(x,y) \end{pmatrix} = - \begin{pmatrix} c_k + H_k x_{\mathcal{I}_k} - A_k^T y + \rho_k A_k^T (A_k x_{\mathcal{I}_k} - b) \\ A_k x_{\mathcal{I}_k} - b \end{pmatrix}. \tag{3.13}$$

The Newton system that results from the linearization of (3.13) about $(\widetilde{x}_k, \widetilde{y}_k)$ is

$$\begin{pmatrix} H_k + \rho_k A_k^T A_k & -A_k^T \\ A_k & \end{pmatrix} \begin{pmatrix} \Delta\widehat{x}_{\mathcal{I}_k} \\ \Delta\widehat{y} \end{pmatrix} = - \begin{pmatrix} c_k + H_k \widetilde{x}_k - A_k^T \widetilde{y}_k + \rho_k A_k^T (A_k \widetilde{x}_k - b) \\ A_k \widetilde{x}_k - b \end{pmatrix}. \tag{3.14}$$

Note from (3.14) that $A_k \Delta\widehat{x}_{\mathcal{I}_k} = b - A_k \widetilde{x}_k$. If we substitute this expression into the first equation and rearrange terms, we arrive at a system identical to (2.5).

4 Numerical Results

We implemented the QPFIL algorithm on a subset of medium-scale QPs from the CUTEr [45] test set, with the aim of demonstrating the global convergence and active-set identification properties of QPFIL in practice. Some important efficiency considerations concerning adequate preconditioners for the KKT system are left for future work.

4.1 Implementation Details

The QPFIL algorithm is based on two computational kernels: the gradual minimization of the bound-constrained augmented Lagrangian function (Step 5 of Algorithm 2) and the solution of an EQP (Step 16 of Algorithm 3).

Our implementation uses the bound-constrained solver within TAO [8] (version 1.8.1), which is based on TRON [51], for Step 5 of Algorithm 2. TAO's flexible interface allows a user-defined termination criterion; we use this feature to implement the filter-based termination criterion defined in Steps 18–24 of Algorithm 2.

The EQP in Step 16 of Algorithm 3 is solved by using PETSc [2–4] (version 2.3.1). In particular, we use PETSc's implementation of GMRES with a restart frequency of 300 and an iteration limit of 1000. Other linear solvers can easily be used within the PETSc framework, and specialized linear solvers and preconditioners for KKT systems are currently an active area of research (see, e.g., [31, 43, 61]). We have deliberately chosen the general-purpose solver GMRES because it is readily available within PETSc and because it simplifies our initial implementation.

We compare our implementation with two general-purpose interior-point solvers: KNITRO [13,15,16,66] and LOQO [64]. Although these methods are targeted to general nonlinear optimization problems, both solvers detect whether the problem is a QP and use appropriate algorithmic options. At this stage we are not interested in a direct comparison between QPFIL and these production-quality interior implementations; rather, we are interested in the number of iterations it takes for QPFIL to identify the optimal active set—KNITRO and LOQO serve as relative benchmarks.

All tests were performed on a desktop PC with a 2.5 GHz Intel Pentium 4 processor with 512 KB RAM, running Red Hat Linux version 7.3. Our implementation is compiled using the GNU gcc compiler (version 3.3.5) with the `-O` flag. The source code and makefiles are available from the second author upon request.

4.2 Test Problems

Our test problems are selections from the AMPL versions of the CUTEr test problems [65]. Problems with up to 20,000 variables or constraints were chosen. General inequality constraints were converted into equalities by introducing slack variables. The chosen test problems and their sizes are listed in Table 1.

4.3 Results of Numerical Experiments

We present the results of our experiments in the form of two performance profiles [26]. Performance profiles provide a convenient way to compare the relative performance of different solvers. For each solver s and every problem p we let the performance measure be $m_{s,p}$, and compute

$$r(s,p) := \log_2\left(\frac{m_{s,p}}{\min_p m_{s,p}}\right),$$

the base-two logarithm of the performance of solver s on problem p divided by the best performance. Ordering and plotting $r(s,p)$ shows the percentage of problems for which a

Table 1: Selected CUTEr test QPs. The name of each problem is followed by the number of original and additional slack variables (n) and the number of linear constraints (m).

Name	n	m	Name	n	m	Name	n	m
aug2dc	20200	9996	aug2dcqp	20200	9996	aug2dqp	20192	9996
aug3dcqp	3873	1000	aug3dqp	3873	1000	avgasa	12	6
avgasb	12	6	biggsc4	11	7	blockqp1	2006	1001
blockqp2	2006	1001	blockqp4	2006	1001	cvxqp1	1000	500
dual1	85	1	dual2	96	1	dual3	111	1
dual4	75	1	dualc5	8	1	genhs28	10	8
hatfldh	11	7	hs021	3	1	hs035	4	1
hs044	10	6	hs053	5	3	hs076	7	3
hs118	32	17	hs268	10	5	huestis	10000	2
ksip	1020	1000	liswet11	20002	10000	liswet12	20002	10000
liswet2	20002	10000	liswet7	20002	10000	lotschd	12	7
mosarqp1	3200	700	mosarqp2	1500	600	ncvxqp5	1000	250
ncvxqp6	1000	250	tame	2	1	zecevic2	4	2

given solver performs at most 2^x worse that the best solver. This plot can be interpreted as a probability distribution that a solver is within a given factor of the best solver. The asymptotics on the right show the reliability of each solver.

We use two metrics to compare the solvers: major (i.e., outer) iterations and CPU time. The first metric is the most similar across the solvers: at each outer iteration, QPFIL solves a KKT system that is structurally similar to the KKT system that interior-point methods need to solve at each of their own "major" iterations. The second measure is more dependent on the implementation of the linear algebra. Figure 3 shows the performance profile in terms of major iterations. We note that QPFIL easily outperforms the two interior-point methods (its profile is barely visible in the top left corner). This indicates that our strategy for finding the optimal active set is efficient.

The second performance profile shows the relative performance in terms of CPU time; see Figure 4. Here, the interior-point codes are faster than QPFIL. This disappointing performance of our algorithm can be traced mainly to the poor performance of the GMRES solver on the KKT system (2.5). On a typical run, for example, code profiling indicates that the KKT solver consumes almost 90% of CPU time. This situation is disappointing considering the ease with which we are able to solve the bound-constrained subproblem with a conjugate gradient method. We believe that better preconditioners and iterative solvers specially designed for KKT systems will dramatically improve this performance. More detailed numerical results taking these aspects into account are currently collected on Argonne's parallel computing resources.

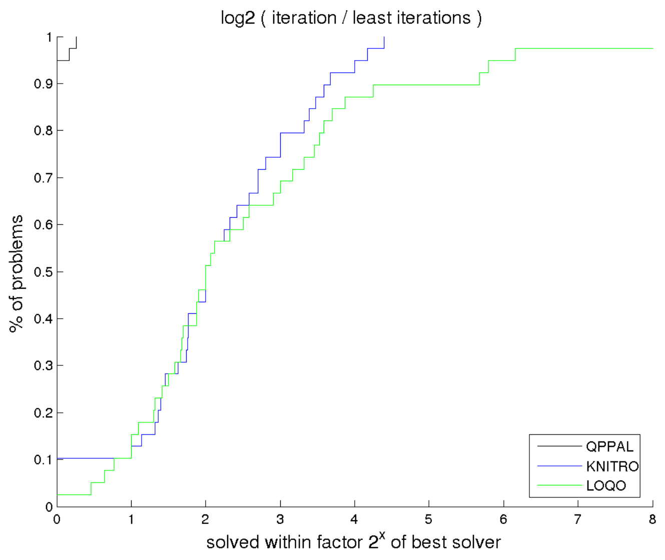

Figure 3: Performance profiles (based on major iterations) for QPFIL versus two interior-point solvers. QPFIL requires the fewest number of KKT solves for about 95% of the test problems.

5 Discussion and Conclusions

We have presented a new active-set method for solving QPs that has the potential for solving very large problems and holds the promise of working efficiently on high-performance architectures. We are encouraged by the speed with which the method identifies a correct active set (see Proposition 3.7). One of the remaining challenges for an efficient implementation is finding a computationally effective way to solve the KKT systems arising from the equality-constrained QP subproblems. In any case, this is the same problem that must be faced by any interior-point implementation—the advantage that we hope to leverage in the QPFIL framework is that the KKT systems are not arbitrarily ill-conditioned.

Two interesting questions remain that we will address in future reports. The first question arises out of Proposition 3.7: Is it possible to simplify the inner iterations further and merely require gradient projection steps until a filter-acceptable point is found? This approach may require a Cauchy-like condition on the inner iteration (which currently is included implicitly by assuming that we perform a few iterations of the minimization of the augmented Lagrangian). Such an approach would have the advantage of removing the need for conjugate gradient iterations involving the Hessian of the augmented Lagrangian, namely, $H + \rho A^T A$, which may be difficult to precondition because of the presence of the term $A^T A$.

The second question concerns the usefulness of the second-order step. If we are far from the minimum, then it may be better to choose the step that adds the largest area to the filter, rather than take a short step in the directions of the second-order point. We plan to investigate this question numerically.

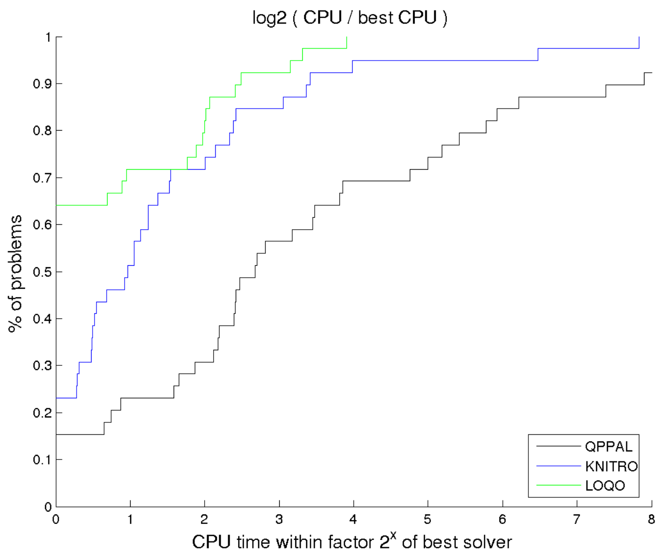

Figure 4: Performance profiles (based on CPU time) for QPFIL versus two interior-point solvers.

Acknowledgments

We are grateful to our summer student Anatoly Eydelzon, who developed an initial version of QPFIL using TAO and PETSC, and to Ewout van den Berg for his careful reading of this paper.

References

[1] M. Anitescu. Optimization-based simulation of nonsmooth rigid multibody dynamics. *Math. Prog.*, 105(1):113–143, 2006.

[2] Satish Balay, Kris Buschelman, Victor Eijkhout, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Barry F. Smith, and Hong Zhang. PETSc Users manual. Technical Report ANL-95/11 (Revision 2.1.5), Argonne National Laboratory, 2004.

[3] Satish Balay, Kris Buschelman, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Barry F. Smith, and Hong Zhang. PETSc Web page, 2001. `http://www.mcs.anl.gov/petsc`.

[4] Satish Balay, William D. Gropp, Lois Curfman McInnes, and Barry F. Smith. Efficient management of parallelism in object oriented numerical software libraries. In E. Arge, A. M. Bruaset, and H. P. Langtangen, editors, *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhauser, 1997.

[5] J. L. Barlow and G. Toraldo. The effect of diagonal scaling on projected gradient methods for bound constrained quadratic programming problems. *Optim. Methods and Software*, 5(3):235–245, 1995.

[6] G. Bashein and M. Enns. Computation of optimal controls by a method combining quasi-linearization and quadratic programming. *Internat. J. Control*, 16(1):177–187, 1972.

[7] S. J. Benson, L. Curfman McInnes, J. J. Moré, and J. Sarich. Scalable algorithms in optimization: Computational experiments. Technical Report ANL/MCS-P1175-0604, Mathematics and Computer Science Division, Argonne National Laboratory, 2004.

[8] Steven J. Benson, Lois Curfman McInnes, Jorge Moré, and Jason Sarich. TAO user manual (revision 1.8). Technical Report ANL/MCS-TM-242, Mathematics and Computer Science Division, Argonne National Laboratory, 2005. `http://www.mcs.anl.gov/tao`.

[9] Luca Bergamaschi, Jacek Gondzio, and Giovanni Zilli. Preconditioning indefinite systems in interior point methods for optimization. *Computational Optimization and Applications*, 28(2):149–171, July 2004.

[10] Dimitri P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods.* Academic Press, New York, 1982.

[11] Dimitri P. Bertsekas. *Nonlinear Programming.* Athena Scientific, Belmont, MA, second edition, 1999.

[12] M. J. Best and J. Kale. Quadratic programming for large-scale portfolio optimization. In J. Keyes, editor, *Financial Services Information Systems*, pages 513–529. CRC Press, 2000.

[13] R. H. Byrd, J. Ch. Gilbert, and J. Nocedal. A trust region method based on interior point techniques for nonlinear programming. *Math. Prog.*, 89:149–185, 2000.

[14] R. H. Byrd, N. I. M. Gould, J. Nocedal, and R. A. Waltz. An algorithm for nonlinear optimization using linear programming and equality constrained subproblems. *Mathematical Programming, Series B*, 100(1):27–48, 2004.

[15] R. H. Byrd, M. E. Hribar, and J. Nocedal. An interior point algorithm for large scale nonlinear programming. *SIAM J. Optim.*, 9(4):877–900, 1999.

[16] R. H. Byrd, J. Nocedal, and R. A. Waltz. Knitro: An integrated package for nonlinear optimization. In G. di Pillo and M. Roma, editors, *Large-Scale Nonlinear Optimization*, pages 35–59. Springer Verlag, 2006.

[17] M. D. Canon and J. H. Eaton. A new algorithm for a class of quadratic programming problems with application to control. *SIAM J. Control*, 4:34–45, 1996.

[18] C.M. Chin and R. Fletcher. On the global convergence of an SLP-filter algorithm that takes EQP steps. *Math. Prog.*, 96(1):161–177, 2003.

[19] T. F. Coleman and L. A. Hulbert. A direct active set algorithm for large sparse quadratic programs with simple bounds. *Mathematical Programming, Series B*, 45(3):373–406, 1989.

[20] T. F. Coleman and Y. Li. A reflective Newton method for minimizing a quadratic function subject to bounds on some of the variables. *SIAM J. Optim.*, 6(4):1040–1058, 1996.

[21] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. Testing a class of methods for solving minimization problems with simple bounds on the variables. *Math. Comp.*, 50:399–430, 1988.

[22] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds. *SIAM J. Numer. Anal.*, 28(2):545–572, April 1991.

[23] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *LANCELOT: A Fortran package for large-scale nonlinear optimization (Release A).* Springer Verlag, Heidelberg, 1992.

[24] F. Delbos and J. Gilbert. Global linear convergence of an augmented lagrangian algorithm for solving convex quadratic optimization problems. *J. Convex Anal.*, 12:45–69, 2005.

[25] M. A. Diniz-Ehrhardt, M. A. Gomes-Ruggiero, and S. A. Santos. Numerical analysis of leaving-face parameters in bound-constrained quadratic minimization. Technical Report 52/98, Department of Applied Mathematics, IMECC-UNICAMP, Campinas, Brasil, 1998.

[26] Elizabeth D. Dolan and Jorge Moré. Benchmarking optimization software with performance profiles. *Math. Prog.*, 91(2):201–213, 2002.

[27] Z. Dostál. Box constrained quadratic programming with controlled precision of auxiliary problems and applications. *Z. Angew. Math. Mech.*, 76(S3):413–414, 1996.

[28] Z. Dostál. Box constrained quadratic programming with proportioning and projections. *SIAM J. Optim.*, 7(3):871–887, 1997.

[29] Z. Dostál, A. Friedlander, and S. A. Santos. Adaptive precision control in quadratic programming with simple bounds and/or equality constraints. In R. De Leone, A. Murli, P. M. Pardalos, and G. Toraldo, editors, *High Performance Algorithms and Software in Nonlinear Optimization*, pages 161–173, Dordrecht, The Netherlands, 1998. Kluwer Academic Publishers.

[30] Z. Dostál, A. Friedlander, and S. A. Santos. Augmented Lagrangians with adaptive precision control for quadratic programming with equality constraints. *Computational Optimization and Applications*, 14(1):37–53, 1999.

[31] Iain S. Duff. Ma57—a code for the solution of sparse symmetric definite and indefinite systems. *ACM Trans. Math. Softw.*, 30(2):118–144, 2004.

[32] K. A. Fegley, S. Blum, J. O. Bergholm, A. J. Calise, J. E. Marowitz, G. Porcelli, and L. P. Sinha. Stochastic and deterministic design and control via linear and quadratic programming. *IEEE Trans. Automat. Control*, AC-16(6):759–766, 1971.

[33] R. Fletcher. A general quadratic programming algorithm. *J. Inst. Math. Appl.*, 7:76–91, 1971.

[34] R. Fletcher. Resolving degeneracy in quadratic programming. *Annals of Operations Research*, 47:307–334, 1993.

[35] R. Fletcher and E. Sainz de la Maza. Nonlinear Programming and nonsmooth Optimization by successive Linear Programming. *Math. Prog.*, 43:235–256, 1989.

[36] R. Fletcher and S. Leyffer. Nonlinear programming without a penalty function. *Math. Prog.*, 91(2):239–269, January 2002.

[37] Anders Forsgren, Philip E. Gill, and Margaret H. Wright. Interior methods for nonlinear optimization. *SIAM Rev.*, 44(4):525–597, 2002.

[38] A. Friedlander and J. M. Martínez. On the maximization of a concave quadratic function with box constraints. *SIAM J. Optim.*, 4:177–192, 1994.

[39] P. E. Gill, W. Murray, and M. A. Saunders. User's guide for QPOPT 1.0: A Fortran package for quadratic programming. Technical Report SOL 95-4, Department of Operations Research, Stanford University, Stanford, 1995.

[40] P. E. Gill, W. Murray, and M. A. Saunders. User's guide for SQOPT 5.3: A Fortran package for large-scale linear and quadratic programming. Technical Report NA 97-4, Department of Mathematics, University of California, San Diego, 1997.

[41] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. Inertia-controlling methods for general quadratic programming. *SIAM Rev.*, 33(1):1–36, 1991.

[42] Philip E. Gill, Walter Murray, Dulce B. Ponceleón, and Michael A. Saunders. Preconditioners for indefinite systems arising in optimization. *SIAM J. Matrix Anal. Appl.*, 13(1):292–311, 1992.

[43] G.H. Golub, C. Greif, and J.M. Varah. An algebraic analysis of a block diagonal preconditioner for saddle point systems. *SIAM J. Matrix Anal. Appl.*, 27(3):779–792, 2006.

[44] N. I. M. Gould, S. Leyffer, and Ph. L. Toint. A multidimensional filter algorithm for nonlinear equations and nonlinear least squares. *SIAM J. Optim.*, 15(1):17–38, 2004.

[45] N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEr and SifDec: A constrained and unconstrained testing environment, revisited. *ACM Trans. Math. Software*, 29(4):373–394, December 2003.

[46] N. I. M. Gould and Ph. L. Toint. SQP methods for large-scale nonlinear programming. In M. J. D. Powell and S. Scholtes, editors, *System Modelling and Optimization, Methods, Theory and Applications*, pages 149–178. Kluwer Academic Publishers, 2000.

[47] N. I. M. Gould and Ph. L. Toint. An iterative working-set method for large-scale non-convex quadratic programming. Technical Report RAL-TR-2001–026, Rutherford Appleton Laboratory, Chilton, Oxfordshire, England, 2001.

[48] N. I. M. Gould and Ph. L. Toint. Numerical methods for large-scale non-convex quadratic programming. In A. H. Siddiqi and M. Kocvara, editors, *Trends in Industrial and Applied Mathematics*, pages 149–179. Kluwer Academic Publishers, 2002.

[49] G. D. Hart and M. Anitescu. A hard-constraint time-stepping approach for rigid multi-body dynamics with joints, contact, and friction. In *Proceedings of the ACM 2003 Tapia Conference for Diversity in Computing*, pages 34–40, Atlanta, GA, 2003.

[50] R. M. Larsen. Combining implicit restart and partial reorthogonalization in Lanczos bidiagnalization, 2001. `http://sun.stanford.edu/~rmunk/PROPACK/`.

[51] Ch.-J. Lin and J. J. Moré. Newton's method for large bound-constrained optimization problems. *SIAM J. Optim.*, 9(4):1100–1127, 1999.

[52] K. Malanowski. On application of a quadratic programming procedure to optimal control problems in systems described by parabolic equations. *Control and Cybernetics*, 1(1–2):43–56, 1972.

[53] H. M. Markowitz. The optimization of a quadratic function subject to constraints. *Naval Research Logistics Quarterly*, 3:111–133, 1956.

[54] A. D. Martin. Mathematical programming of portfolio selections. *Management Science*, 1(2):152–166, 1955.

[55] A. Mohri. A computational method for optimal control of a linear system by quadratic programming. *Internat. J. Control*, 11(6):1021–1039, 1970.

[56] J. J. Moré and G. Toraldo. On the solution of quadratic programming problems with bound constraints. *SIAM J. Optim.*, 1(1):93–113, 1991.

[57] Stephen G. Nash and Ariela Sofer. *Linear and Nonlinear Programming.* McGraw-Hill, New York, 1996.

[58] Y. E. Nesterov and A. Nemirovski. *Interior-point polynomial methods in convex programming*, volume 14 of *Studies in Applied Mathematics.* SIAM, Philadelphia, 1994.

[59] J. Nocedal and S.J. Wright. *Numerical Optimization.* Springer Verlag, Berlin, 1999.

[60] R. T. Parry. The application of quadratic programming to the portfolio selection problem: a review. Special Studies Paper 5, Board of Governors of the Federal Reserve System, USA, 1970.

[61] Olaf Schenk and Klaus Gärtner. Solving unsymmetric sparse systems of linear equations with PARDISO. *Future Generation Computer Systems*, 20(3):475–487, April 2004.

[62] Tomomichi Sugihara and Yoshihiko Nakamura. Balanced micro/macro contact model for forward dynamics of rigid multibody. Technical report, Department of Mechano-informatics, University of Tokyo, Tokyo, 2006.

[63] M. Ulbrich, S. Ulbrich, and L.N. Vicente. A globally convergent primal-dual interior-point filter method for nonlinear programming. *Math. Prog.*, 100(2):379–410, 2004.

[64] R.J. Vanderbei and D.F. Shanno. An interior point algorithm for nonconvex nonlinear programming. *COAP*, 13:231–252, 1999.

[65] Robert Vanderbei. Benchmarks for nonlinear optimization, December 2002. `http://www.princeton.edu/~rvdb/bench.html`.

[66] R. A. Waltz and T. D. Plantenga. *KNITRO 5.0 User's Manual.* Ziena Optimization, Inc., Evanston, IL, February 2006.

[67] J. Weston and W. Barenek. Programming investment portfolio construction. *Analysts Journal*, 11:51–55, 1955.